Migrating UNIX SysAdmin Tools to Mac OS X

Session 613

Migrating UNIX SysAdmin Tools to Mac OS X

Skip Levens
Server Technology Evangelist
slevens@apple.com

Bridging Mac OS 9 and UNIX With Mac OS X

Kevin A. Boyd
Mac OS X Deployment Coordinator
University of Michigan
kboyd@umich.edu

Kevin A. Boyd–Technical Background

- IT Systems Administrator at the University of Michigan
  - An institution with a highly decentralized technology infrastructure
- Steering Committee for MacOSXLabs.org

University of Michigan

- UM allows individual academic units total freedom over their computing environments, presenting administrators and consultants with a wide array of technology integration issues

LDAP
Kerberos
OpenAFS
ActiveDirectory
Netware
AppleTalk
SSH,SCP,FTP

Mac OS 9 Experience at UM

- Using 3rd party tools and custom applications, Mac OS 9 machines were easy to manage, but difficult to integrate
- When Mac OS 9 was the only solution, machines using flavors of UNIX were easy to integrate but comparably difficult to manage

University of Michigan

- Mac OS X was seen as an opportunity to create tools that would make integrating and managing machines easy
- UM has leveraged its Mac OS 9 and UNIX expertise to create new lab management tools for Mac OS X
- Challenge: Keep the identified benefits of Mac OS 9 experience yet leverage power of Mac OS X to build lab management tools for all users

Case Study:
University of Michigan:
A Journey From Mac OS 9 To Mac OS X

The University Is The Enterprise

- The University of Michigan defines a typical enterprise computing environment
- The freedom and complexity at every level presents an unlimited array of potential problems and solutions
  - Hardware, Software, and Data
  - People and Procedures
- **Case Study Focus:**
  University of Michigan School of Art and Design

Mac OS 9 vs. UNIX

	BENEFITS:	DRAWBACKS:
Mac OS 9	* Assimilator: loadset management tool * Custom loadsets * Fast configuration	* Did not scale * No Kerberos * AFS support via netatalk * Limited multi-user support
UNIX	* All flavors of UNIX on campus * LDAP, AFS, Kerberos, SAMBA, * Must integrate with existing environment * Share command config files across flavors of UNIX	* Custom loadsets via CLI only * More difficult to use

How To Create Tools Administrators Will Use

SysAdmin Tools: Identify Expectations and Benefits of Your Audience

- Die-Hard Mac OS 9 Administrators
- Hardcore UNIX Administrators
- Savvy Mac OS X Administrators
- Each Audience has its own expectations from your software, but you can fill the needs of all groups with a single application

Die-Hard Mac OS 9 Administrators Want:

- An intuitive, easy to use interface
- On screen help (ToolTips)
- High degree of Mac OS X integration
- No ‘learning curve’

Hardcore UNIX Administrators Want:

- Command Line access
- Man pages or online help
- Portable UNIX calls
- Source Code, if possible

Savvy Mac OS X Administrators Want:

- Everything That Die-Hard Mac OS 9 Administrators want
- Everything That Hardcore UNIX Administrators want
- Portability of applications or multiple versions of your application to support multiple versions of Mac OS X

Case Study:
RsyncX and RsyncXCD

Give The People What They Want

- Candidate for Conversion: RsyncX
- RsyncX Mission
- Software Modelling:
  - RsyncX
  - RsyncXCD
  - RsyncX Package Tools
- Solves the Loadset Problem

RsyncX Development Cycle

- Identify Audiences: UNIX, Mac OS 9, and Mac OS X System Administrators
- Gather input: email, the web, and hands-on, face-to-face meetings
- Develop in three parts: UNIX underpinnings, User Interface, and Documentation
- No shortcuts

RsyncX Script Generator

- The Script Generator creates command files that automate your file management activities
- These command files can be activated from the GUI with a simple double-click
- These command files can be viewed in any text editor or Terminal window
- These command files are highly portable–the commands used in a script can be used on any other UNIX platform with the same utilities installed

Hitting All Target Audiences and More

- The RsyncX Project addresses the basic needs of all three target audiences
- They are “works in progress”–as the user interface and UNIX underpinnings improve, the target audience widens
- Documenting and providing usage examples could cause your audience to widen beyond the three SysAdmin groups targeted here

RsyncX–A Piece In The Puzzle

- As development cycles continue, new uses are always found for RsyncX
- Adding functionality during the development cycle will keep your software relevant
- “There Is No Such Thing As Bad Press”–Responding to your audience with help and updates will only improve your application
- Use the Developer Tools CD–it will insure that your application keeps Mac OS X a cooperative, well-behaved network citizen

Migrating UNIX SysAdmin Tools to Mac OS X

Porting UNIX Command Line Tools

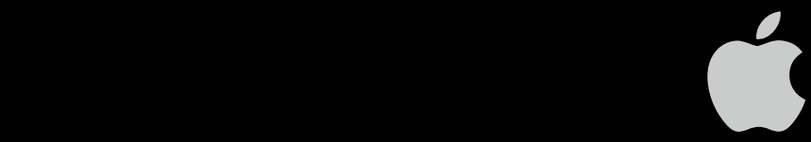
Waqar Malik
DTS Software Engineer
wmalik@apple.com

Introduction

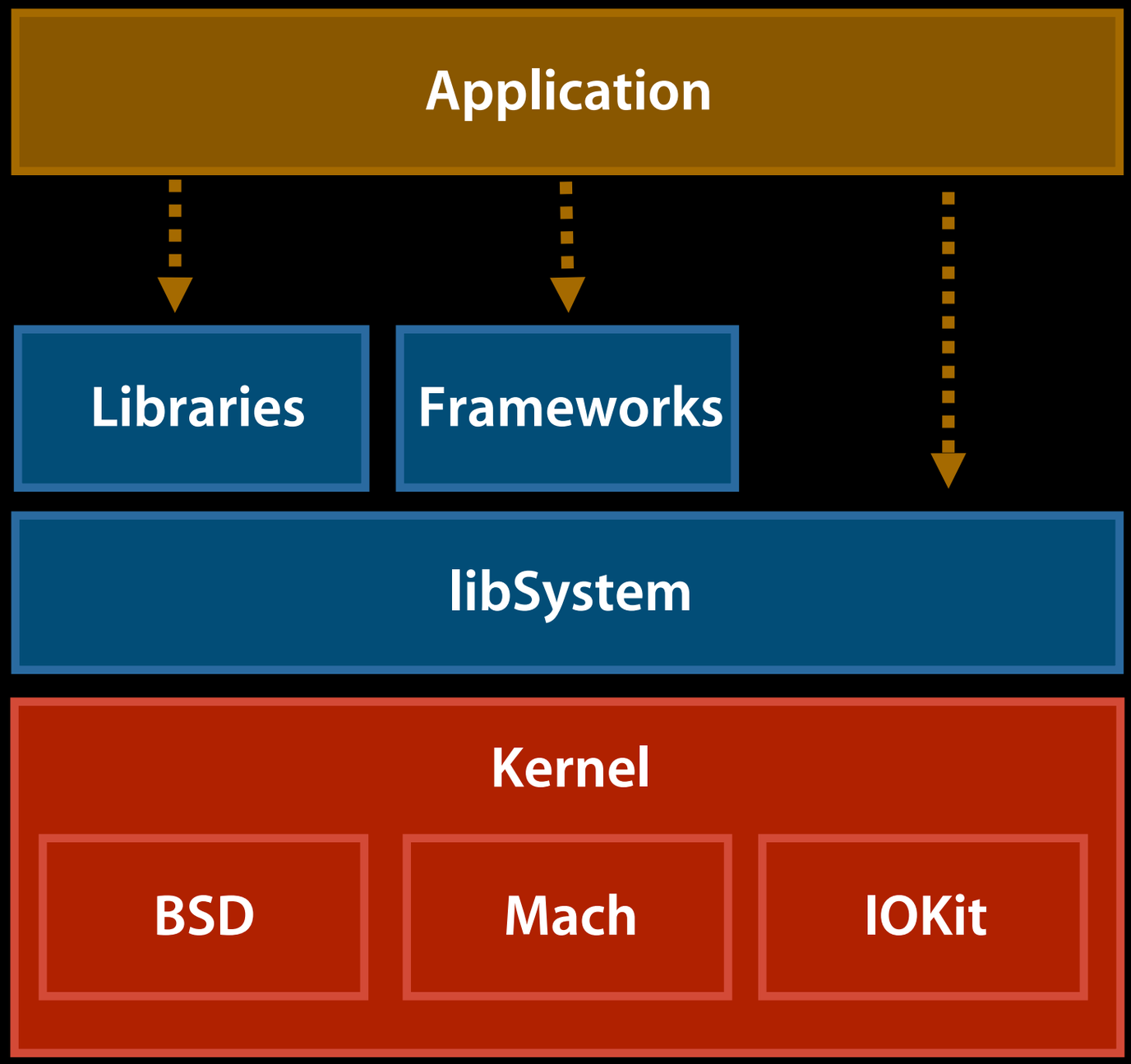
- Port UNIX command line applications to Mac OS X
- Background needed to understand the Mac OS X
- Information on areas to investigate

What You'll Learn

- Differences between Mac OS X and other UNIX systems
  - Supported APIs
  - Compatibility APIs
  - Different APIs
- How easy it can be to port UNIX applications
- What to do when it isn't easy



Basic Application Diagram

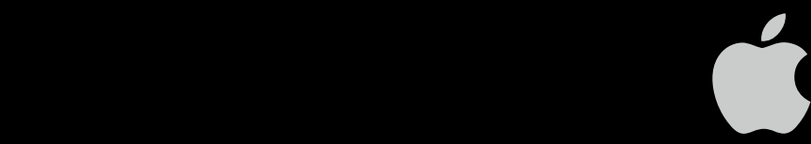


Tools You will Need

- BSD Subsystem
  - X11 for Mac OS X
- Developer Tools
  - BSD SDK
  - X11 SDK

Development Environment

- Mostly GNU toolchain
  - gcc
  - gdb
  - make
- Xcode
- Interface Builder
- MallocDebug
- Various other performance-tuning tools



Choosing a Compiler

- Three different versions
  - GCC 3.3
  - GCC 3.1
  - GCC 2.95
- Selecting a compiler
  - Using `gcc_select`
  - Xcode
  - Full path

Before Doing It Yourself

- Check if someone's done it before
  - DarwinPorts (included in Panther!)
  - Fink (SourceForge)
  - Mac OS X Downloads
- If not, find an example to follow
- Afterwards, submit your port for others to use

Porting Scripts

- Lots of built-in languages
  - Shell (bash, tcsh, zsh)
  - Perl
  - TCL
  - Python
  - Ruby
- `/bin/sh` is `bash`
- AppleScript (like a shell for GUI apps)
- Porting Is Easy–but filenames can be tricky

Primary File System

- There are many file systems available for Mac OS X
- HFS+
  - Case Insensitive, Case Preserving
  - Resource Forks
  - Journaling
- UFS
  - Case Sensitive
  - Sparse Files (not universal)

Traditional Mac Users Issues

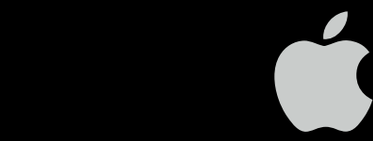
- Tend to use spaces in filenames
- Tend not to use filename extensions (e.g., foo.txt)
  - Use Type and Creator codes instead
- Resource Forks
- Aliases vs. Links

Libraries

- Mach-O binary format
  - Versioning
  - Plug-ins
  - Bundles vs. Libraries
- .dylib extension
- Static linking rarely used
- Frameworks
- libSystem

Bundles

- Directory that stores executable code and software resources related to that code
- Pros
  - Fat binaries (ppc, x86, Mach-O, CFM)
  - Localization
  - Easy installation/un-installation
- Examples
  - Applications, Frameworks, compound documents, etc.



Frameworks

- Bundle based alternative to the UNIX hierarchy
- Contains libraries, headers, resources, and documentation
- Live in the following locations
  - /System/Library/Frameworks
  - /Library/Frameworks
  - /Network/Library/Frameworks
  - ~/Library/Frameworks
  - Application.app/Contents/Frameworks

Two-Level Namespace

- How references to symbols are resolved
- Resolution done at static link time when image (application, bundle, framework) is built
- Allows different global symbols with same name
- Default in Mac OS X 10.1 and later

Runtime Issues

- Configuration and Resource Files
  - Open Directory for system preferences
  - CF/NSPreferences for user preferences
- System startup
  - `/System/Library/StartupItems`
  - `/Library/StartupItems`
  - Bundle based, but not CFBundle
  - Composed of an executable and a plist
  - Localizable

Prebinding

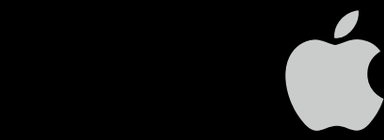
- Benefits
  - Decreases run-time startup cost
- Issues
  - Modifies executables and libraries
  - Confuses applications (backup, security)

Authentication/Authorization

- Security framework
  - Apple's preferred API
  - Rights based system
  - Based on open standard CDSA
- PAM available for compatibility purposes

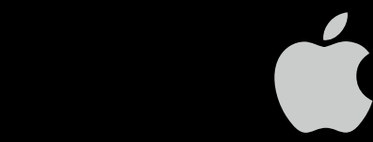
GUI Options

- Cocoa
- Carbon
- Java/Swing
- X11
- Tcl/Tk (X11 or Aqua)
- Qt (now Open Source for Mac OS X)



Multimedia

- Quicktime for high level mixed media
- CoreAudio/CoreMidi for low level audio
- Benefits
  - Native multi-channel audio with plug-in support
  - Native MIDI
  - Built in USB class driver compliant with the USB audio specification
- Cons
  - You have to rewrite your drivers



Device Drivers

- I/O Kit
- Embedded C++
- OO Design
- Use GCC 2.95 to be compatible for previous versions

Internationalization

- Use CoreFoundation
  - CFString
  - CFLocale
  - Unicode
  - Apple recommended
- wchar_t now supported

Pre-compiled Headers

- Apple's preprocessor supports pre-compiled headers
- Some constructs are not supported
- Use `-no-cpp-precomp` to switch to GNU preprocessor
- Do not use `-traditional` to turn off pre-compiled headers

Linker

- Apple's Own linker
- Single Pass
- Lazy Initialization of Variables, important for C++
- Prebinding
- Use `-dynamic` instead of `-shared`

What's New In Panther?

- wchar, libiconv, ICU
- Asynchronous I/O (aio_*)
- poll (scheduled)
- dlopen/dlclose (scheduled)
- PortsManager GUI

Leveraging Apple's Developer Tools to Extend Your Scripts

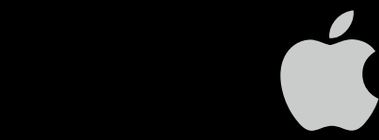
Mike Bombich
Systems Engineer
bombich@apple.com

Apple Developer Tools + Your Scripts

- Package Maker
- Property List Editor
- AppleScript Studio
- Project Builder/Interface Builder
- IconComposer

AppleScript Studio

- AppleScript
- Project Builder
- Interface Builder
- Mac OS 10.1.2+
- Three project templates in Project Builder
- Over 30 sample projects



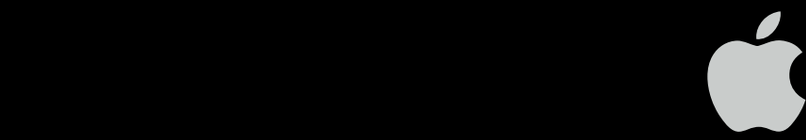
Developing a GUI for a Shell Script

- First develop your algorithm in the CLI
- Make sure that everything will work from within the shell script
- Generalize your shell script–make sure it functions in every possible scenario for your environment
- Anticipate input from non-Unix users

```
% Enter the path to the backup volume:
> /Volumes/Macintosh HD
/bin/mv: “Volumes/Macintosh” not found
```

Project Builder and Interface Builder

- Create a project in Project Builder
  - AppleScript Application (most common)
  - AppleScript Document-based application
  - AppleScript Droplet Application
- Mock up an interface in Interface Builder
  - Decide which interface elements will provide the best form of user input
  - How will you report results/progress to the user?



Attach Your Script Parts to the GUI

- Use AppleScript's "do shell script" function to implement your script's functionality
- GUI apps are not linear like shell scripts, attach specific scripts to individual user actions
- You will probably use global variables more frequently than you are accustomed to

Other Things to Keep In Mind

- Consider security issues:
  - Does your script require admin/root privileges?
  - How will you retrieve that password?
- Would preferences suit this utility well?
  - Use AppleScript Studio's "user-defaults" object to create and reference preferences
- Learn about your environment variables and $PWD

Case Study:
Remote Image Mounter

Problem: Share a library of CDs

- You've been tasked with maintaining a library of a couple hundred CDs
- CDs become scratched, damaged, or lost, therefore you have to maintain backups
- Someone has to manage a checkout system
- CDs are only available to one person at a time

Solution: Share a Library of CD Images

- Instead of sharing the physical discs, you could instead create disk images of the CDs and make them available for remote access using Apple's Disk Images framework
- First, create the disk images and make them available on an Apache web server
- Create a shell script to mount the images:

```
hdiutil mount http://xserve.apple.com/cdlib/disk1.dmg
```

Demo

Another Example

Carbon Copy Cloner

For More Information

Reference Library

Documentation

Inside Mac OS X: System Overview

http://developer.apple.com/techpubs/macMac OS X/Essentials/SystemOverview/index.html

UNIX Porting Guide

http://developer.apple.com/techpubs/macMac OS X/Darwin/GettingStarted/PortingUNIX/index.html

Reference Library (Cont.)

Technical Q&As

QA1026: Calling AppleScript from an Application
http://developer.apple.com/qa/qa2001/qa1026.html

Source Code

Remote Image Mounter
http://www.bombich.com/software/wwdc.html

Who to Contact

Skip Levens
Server Technology Evangelist
slevens@apple.com

Waqar Malik
DTS Software Engineer
wmalik@apple.com

Mike Bombich
Systems Engineer
bombich@apple.com

http://developer.apple.com/wwdc2003/urls.html

Who to Contact

Kevin A. Boyd
Mac OS X Deployment Coordinator
kboyd@umich.edu

RsyncX Development Team
rsyncxdev@umich.edu

http://developer.apple.com/wwdc2003/urls.html

Roadmap

301 Apple Tools for the UNIX Developer	Russian Hill Tues., 9:00am
302 Introduction to Developer Tools Suite	The Presidio Tues., 3:30pm
304 GCC in Depth	The Mission Wed., 3:30pm
308 Debugging Services for Mac OS X	The Presidio Thur., 3:30pm

Reference Library

Documentation

- Mac OS X In Lab Environments
  - http://macMac OS Xlabs.org
- RsyncX
  - http://www.macMac OS Xlabs.org/rsyncx/rsyncx.html

Reference Library

Source Code

- rsync_hfs
  - http://www.opendarwin.org

Reference Library

Other Sessions about AppleScript Studio

306 AppleScript Studio	Pacific Heights Wed., 5:00pm
623 AppleScript for SysAdmins	Pacific Heights Fri., 2:00pm

Q&A

http://developer.apple.com/wwdc2003/urls.html